PENDING CLAIMS

Sub C1

1. (Unamended) A small footprint device comprising:

   a. at least one processing element;

   b. memory, and

   c. a context barrier, using said memory and running on said processing element, for isolating program modules from one another.

25. (Unamended) The small footprint device of claim 1 in which said at least one processing element is a virtual machine running on a processor.

26. (Unamended) The small footprint device of claim 25 in which said virtual machine runs on top of a card operating system.

27. (Unamended) The small footprint device of claim 1 in which groups of one or more program modules are run in separate contexts.

28. (Unamended) The small footprint device of claim 27 in which the context barrier is configured to prevent access from one program module to a program module in a different context.

29. (Unamended) The small footprint device of claim 27 in which said context barrier allocates separate respective name spaces for each context.

30. (Unamended) The small footprint device of claim 27 in which said context barrier allocates separate respective memory spaces for each context.

31. (Unamended) The small footprint device of claim 1 in which at least one program module comprises a plurality of applets.

32. (Unamended) The small footprint device of claim 1 in which said context barrier enforces at least one security check on at least one of principal, object or action to prevent access from a principal in one context to an object in a different context.

33. (Unamended) The small footprint device of claim 32 in which groups of one or more program modules are run in separate contexts.

34. (Unamended) The small footprint device of claim 33 in which the context barrier prevents access from one program module to a different program module.

35. (Unamended) The small footprint device of claim 33 in which at least one security check is based on partial name agreement between a principal and an object.

36. (Unamended) The small footprint device of claim 33 in which at least one security check is based on memory space agreement between a principal and an object.

37. (Unamended) A method of operating a small footprint device, comprising the step of preventing access from one program module to a different program modules using a context barrier.

38. (Unamended) The method of claim 37 in which the context barrier is implemented using a virtual machine.

39. (Unamended) The small footprint device of claim 38 in which groups of one or more program modules are run in separate contexts.

40. (Unamended) The method of claim 39 in which the context barrier prevents access from one program module to a different program module.

41. (Unamended) The method of claim 40 in which the context barrier will not permit a principal to access an object unless both principal and object are part of the same name space.

42. (Unamended) The method of claim 39 in which the context barrier will not permit a principal to access an object unless both principal and object are part of the same memory space.

43. (Unamended) The method of claim 37 in which the context barrier will not permit a principal to perform an action on an object unless both principal and object are part of the same context.

44. (Unamended) The method of claim 43 in which the context barrier will permit a principal to perform an action on an object when they are not part of the same context if the principal is authorized to perform the action on the object.

45. (Unamended) The method of claim 44 in which the principal is authorized if it passes at least one security check.

46. (Unamended) The method of claim 45 in which said at least one security check is one of a plurality of security checks.

47. (Unamended) The method of claim 44 in which, if a principal in a first context is authorized to perforin one or more actions on an object in a second context, when the action is performed it will execute within the second context.

48. (Unamended) The method of claim 47 in which, when one or more actions are authorized in the second context, subsequent actions will be authorized based on executing in the second context, and a principal in the second context will be able to access objects in the second context.

49. (Unamended) The method of claim 48 in which, when one or more actions complete in the second context, execution will return to the first context.

50. (Unamended) The method of claim 47 in which, when action is undertaken in the second context that requires access to an object in a third context, the action will execute within the third context.

51. (Unamended) The method of claim 50 in which switches to a new context will occur any time action is authorized on an object in a new context.

52. (Unamended) A computer program product, comprising:

    a. a memory medium; and

    b. a computer controlling element comprising instructions for implementing a context barrier on a small footprint device.

53. (Unamended) The computer program product of claim 52 in which said memory medium is a carrier wave.

54. (Unamended) A computer program product, comprising:

    a. a memory medium; and

    b. a computer controlling element comprising instructions for separating a plurality of programs on a small footprint device by running them in respective contexts.

55. (Unamended) The computer program product of claim 54 in which said mem . ory medium is a carrier wave.

Sub C1

56. (Unamended) A carrier wave carrying instructions for implementing a context barrier on a small footprint device over a communications link.

57. (Unamended) A carrier wave carrying instructions over a communications link for separating a plurality of programs on a small footprint device by running them in respective contexts.

58. (Unamended) A method of shipping code over a network, comprising the step of transmitting a block of code from a server, said block of code comprising instructions over a communications link for separating a plurality of programs on a small footprint device by running them in respective contexts.